 # import all packages and set plots to be embedded inline

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Load DataSet

```
df=pd.read_csv("BankNote_Authentication.csv")
df.head()
```

	variance	skewness	curtosis	entropy	class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
2	3.86600	-2.6383	1.9242	0.10645	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0

Data Assessing

```
df.info()
```

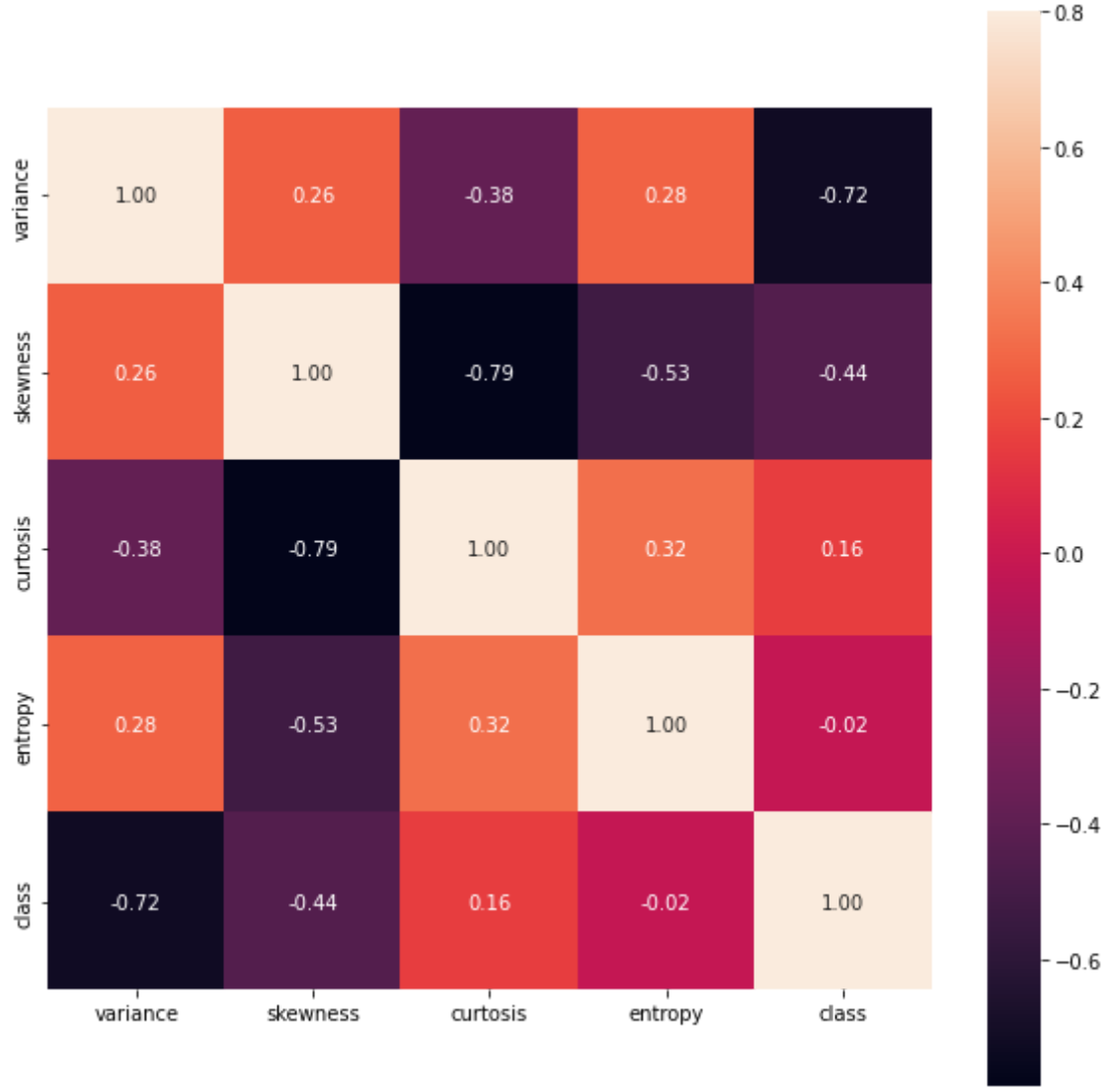
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   variance  1372 non-null   float64
 1   skewness  1372 non-null   float64
 2   curtosis  1372 non-null   float64
 3   entropy   1372 non-null   float64
 4   class     1372 non-null   int64  
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

```
df.describe()
```

	variance	skewness	curtosis	entropy	class
count	1372.000000	1372.000000	1372.000000	1372.000000	1372.000000
mean	0.433735	1.922353	1.397627	-1.191657	0.444606
std	2.842763	5.869047	4.310030	2.101013	0.497103
min	-7.042100	-13.773100	-5.286100	-8.548200	0.000000
25%	-1.773000	-1.708200	-1.574975	-2.413450	0.000000

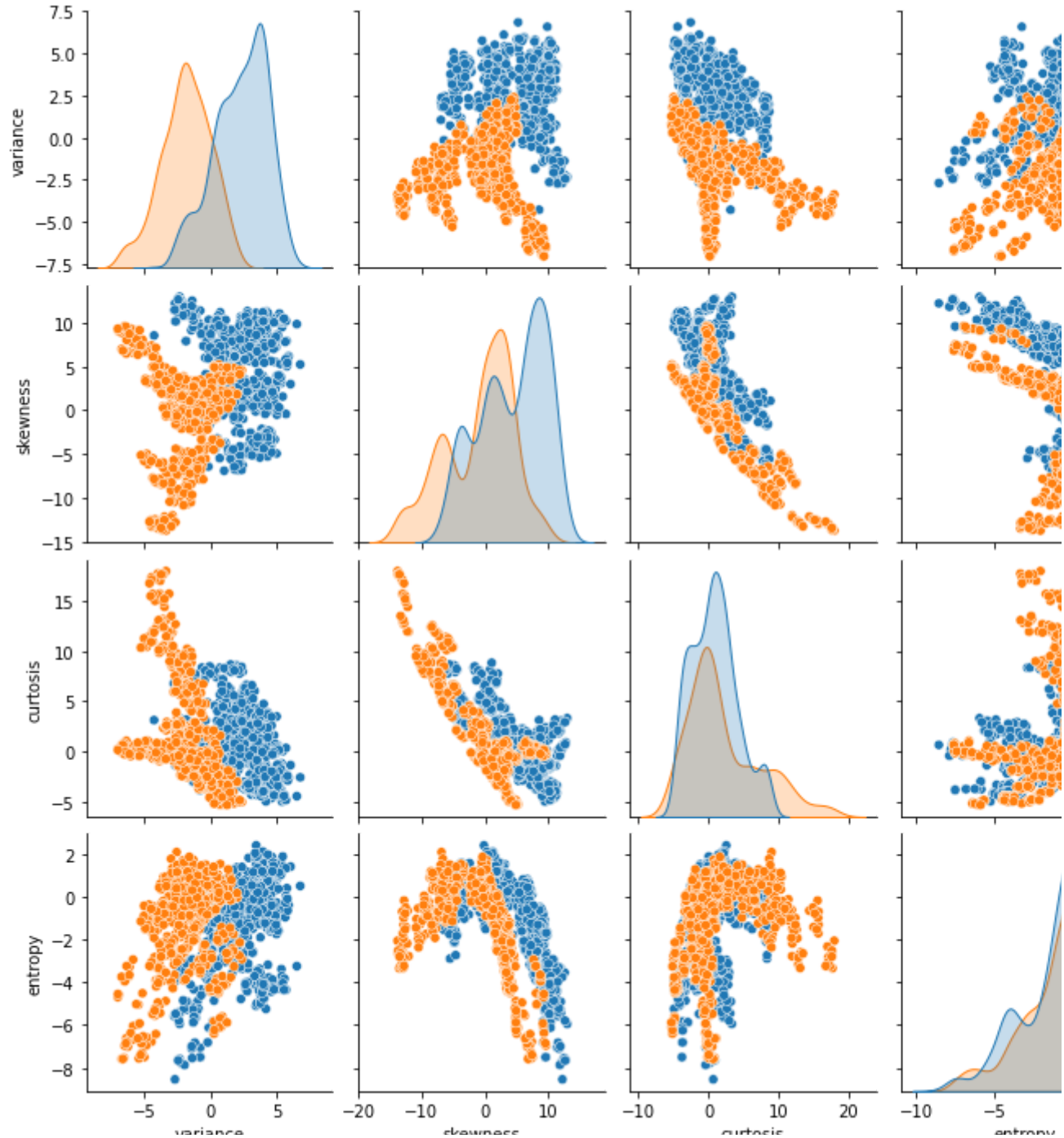
Data Visualization

```
corrmat=df.corr()
f,ax=plt.subplots(figsize=(10,10))
sns.heatmap(corrmat,vmax=0.8 ,square=True ,annot=True,fmt='.2f'
plt.show()
```




```
sns.pairplot(df ,hue='class')
```

```
<seaborn.axisgrid.PairGrid at 0x7f079edaed90>
```



Train Test Split Using Sklearn

```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
```

DecisionTreeClassifications

```
from sklearn.tree import DecisionTreeClassifier
DTC=DecisionTreeClassifier()
DTC
```

```
DecisionTreeClassifier()
```

```
DTC.fit(x_train,y_train)
```

```
DecisionTreeClassifier()
```

```
y_predict=DTC.predict(x_test)
y_predict
```

```
array([1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0,
       0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1,
       0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0])
```

Evaluate the Clustering Model

```
from sklearn.metrics import classification_report,confusion_mat
print(classification_report(y_test,y_predict))
```

```
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       155
           1       0.98      0.98      0.98       120

    accuracy                           0.98       275
   macro avg       0.98      0.98      0.98       275
weighted avg       0.98      0.98      0.98       275
```

```
CM=(confusion_matrix(y_test,y_predict))
print(pd.DataFrame(CM))
```

```
     0    1
0  152    3
1    2  118
```

```
print(accuracy_score(y_test,y_predict))
```

```
0.9818181818181818
```

** Decision Tree **

```
DTC.feature_importances_
```

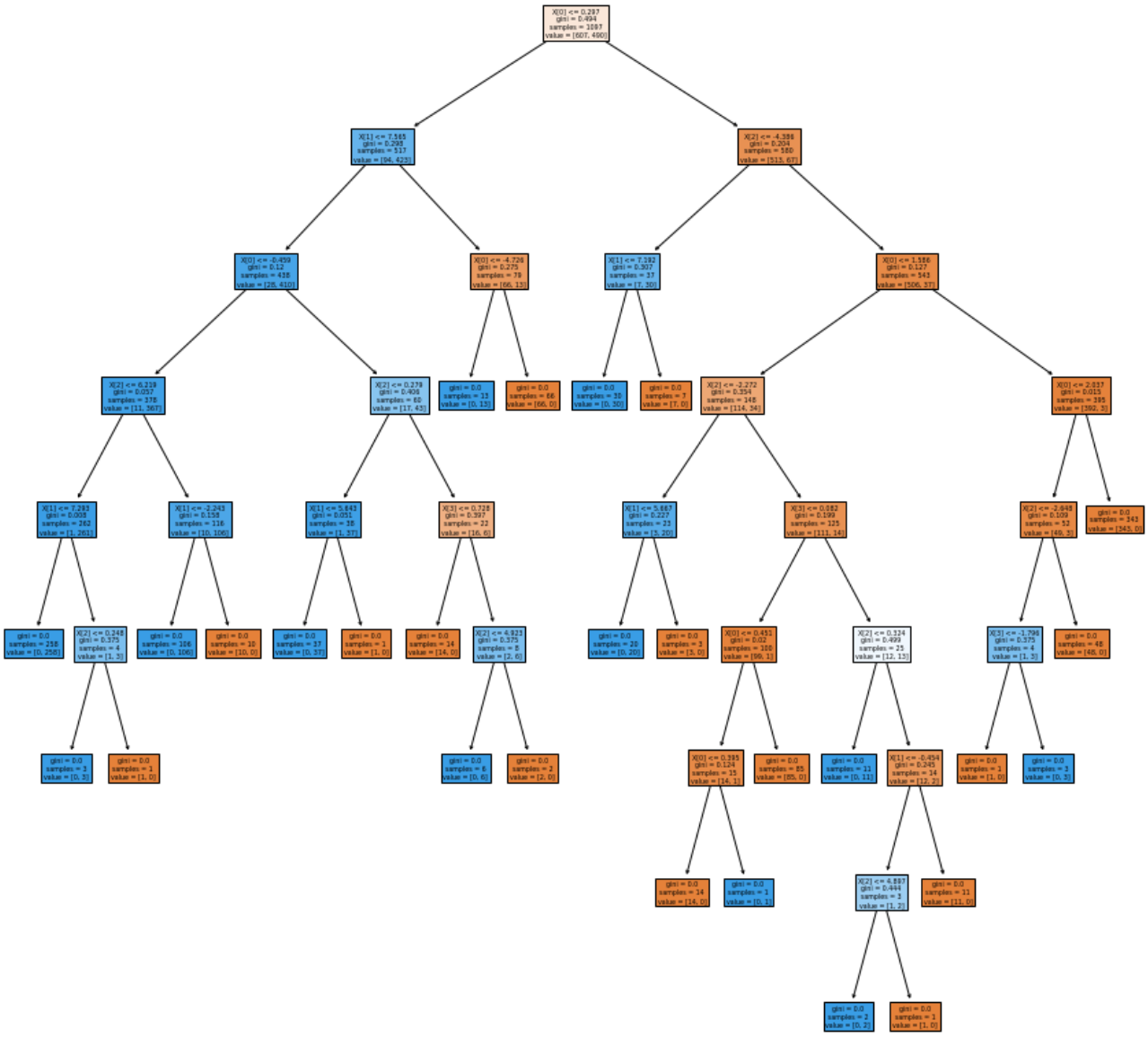
```
array([0.5739719 , 0.21955518, 0.17395853, 0.03251438])
```

```
DTC.feature_importances_
```

```
array([0.5739719 , 0.21955518, 0.17395853, 0.03251438])
```

```
from sklearn import tree
plt.figure(figsize=(15,15))
print(tree.plot_tree(DTC ,filled=True))
```

```
[Text(0.5018115942028986, 0.9444444444444444, 'X[0] <= 0.297\ngini = 0.494\nsamples =
```



✓ 3s completed at 11:04 PM